



ISSM Workshop 2016

Ice Sheet System Model Inverse Methods

Mathieu MORLIGHEM¹

¹University of California, Irvine



Outline

① Introduction

- Model equations
- Inverse problem
- Algorithm

② Hands on 1 (ice rigidity)

- Initial state
- Inversion

③ Hands on 2 (friction)



Ice flow model equations

Field equations

- ① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

- ② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

- ③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu\dot{\boldsymbol{\epsilon}} \quad \mu = \frac{B}{2\dot{\varepsilon}_e^{1-1/n}} \quad (3)$$

Ice flow model equations

Field equations

- ① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

- ② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

- ③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu\dot{\boldsymbol{\varepsilon}} \quad \mu = \frac{B}{2\varepsilon_e^{1-1/n}} \quad (3)$$

Boundary conditions

Ice/Air interface: Free surface

$$\boldsymbol{\sigma} \cdot \mathbf{n} = P_{atm} \mathbf{n} \simeq \mathbf{0} \quad \text{on } \Gamma_s$$

Ice/Ocean interface: water pressure

$$\boldsymbol{\sigma} \cdot \mathbf{n} = P_w \mathbf{n} \quad \text{on } \Gamma_w$$

Ice/Bedrock interface (1): lateral friction

$$(\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} = \mathbf{0} \quad \text{on } \Gamma_b$$

Ice/Bedrock interface (2): impenetrability

$$\mathbf{v} \cdot \mathbf{n} = -\dot{M}_b n_z \quad \text{on } \Gamma_b$$



Ice flow model equations

Field equations

- ① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

- ② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

- ③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu\dot{\boldsymbol{\varepsilon}} \quad \mu = \frac{B}{2\dot{\varepsilon}_e^{1-1/n}} \quad (3)$$

Boundary conditions

Ice/Air interface: Free surface

$$\boldsymbol{\sigma} \cdot \mathbf{n} = P_{atm} \mathbf{n} \simeq \mathbf{0} \quad \text{on } \Gamma_s$$

Ice/Ocean interface: water pressure

$$\boldsymbol{\sigma} \cdot \mathbf{n} = P_w \mathbf{n} \quad \text{on } \Gamma_w$$

Ice/Bedrock interface (1): lateral friction

$$(\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} = \mathbf{0} \quad \text{on } \Gamma_b$$

Ice/Bedrock interface (2): impenetrability

$$\mathbf{v} \cdot \mathbf{n} = -\dot{M}_b n_z \quad \text{on } \Gamma_b$$



Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknowns
- ex: surface velocities derived from InSAR



Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknowns
- ex: surface velocities derived from InSAR

PDE-constrained optimization

Minimize cost function

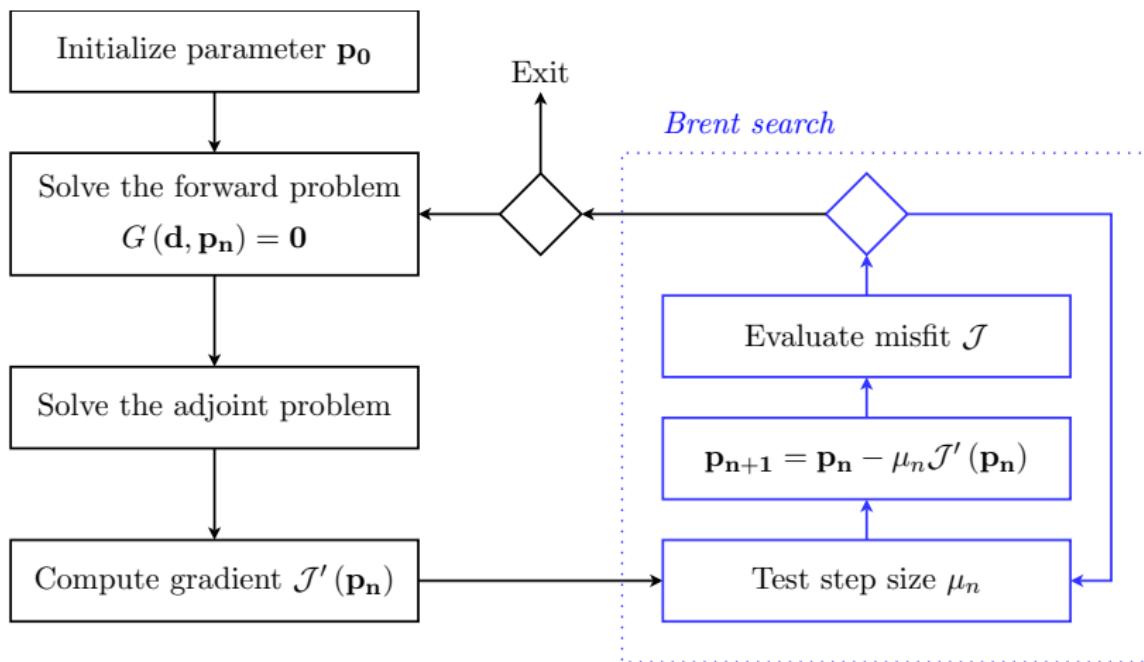
$$\mathcal{J}(\mathbf{v}, \alpha) = \frac{1}{2} \int_{\Gamma_s} \left(v_x - v_x^{obs} \right)^2 + \left(v_y - v_y^{obs} \right)^2 dS + \mathcal{R}(\alpha) \quad (4)$$

Subject to:

$$\begin{aligned}
 \nabla \cdot 2\mu\dot{\epsilon} - \nabla p + \rho\mathbf{g} &= \mathbf{0} && \text{in } \Omega \\
 \nabla \cdot \mathbf{v} &= 0 && \text{in } \Omega \\
 \boldsymbol{\sigma} \cdot \mathbf{n} &= \mathbf{f} && \text{on } \Gamma_s \cup \Gamma_w \\
 (\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} &= \mathbf{0} && \text{on } \Gamma_b \\
 \mathbf{v} \cdot \mathbf{n} &= -\dot{M}_b n_z && \text{on } \Gamma_b
 \end{aligned} \quad (5)$$



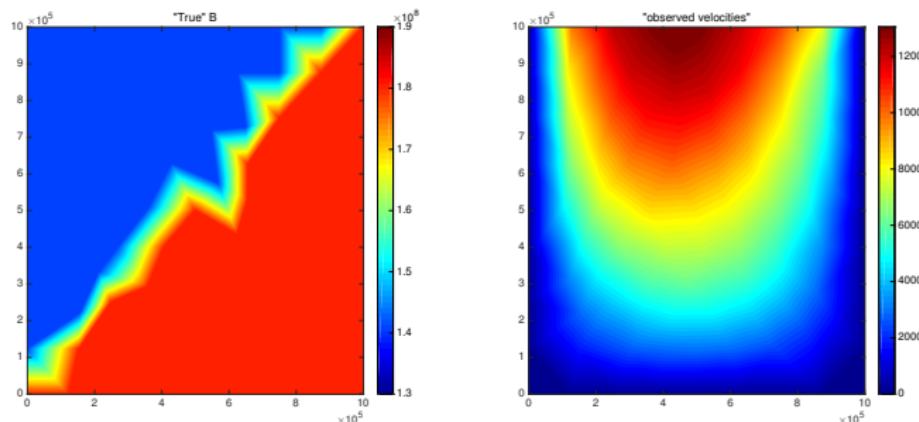
Algorithm of resolution



Hands on 1 (ice rigidity)

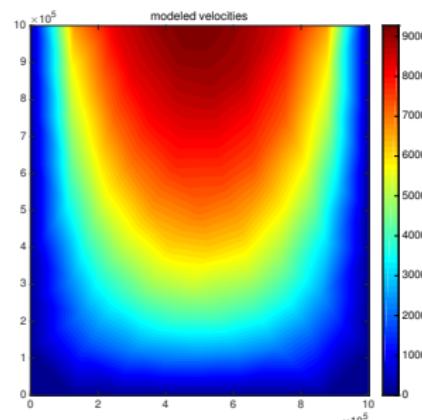
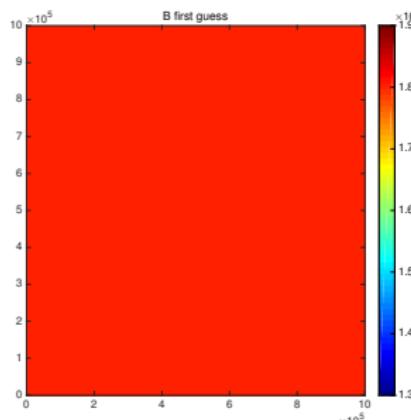
```
3 %Generate observations
4 md = model;
5 md = triangle(md,'DomainOutline.exp',100000);
6 md = setmask(md,'all','');
7 md = parameterize(md,'Square.par');
8 md = setflowequation(md,'SSA','all');
9 md.cluster = generic('np',2);
10 md = solve(md,StressbalanceSolution());
```

```
19 md.materials.rheology_B=1.8*10^8*ones(md.mesh.numberofvertices,1);
20 md.materials.rheology_B(find(md.mesh.x<md.mesh.y))=1.4*10^8;
```



Start from constant hardness

```
16 %Modify rheology, now constant
17 loadmodel('modell.mat');
18 md.materials.rheology_B(:) = 1.8*10^8;
19
20 %results of previous run are taken as observations
21 md.inversion=mlqn3inversion();
22 md.inversion.vx_obs = md.results.StressbalanceSolution.Vx;
23 md.inversion.vy_obs = md.results.StressbalanceSolution.Vy;
24 md.inversion.vel_obs = md.results.StressbalanceSolution.Vel;
25
26 md = solve(md,StressbalanceSolution());
```

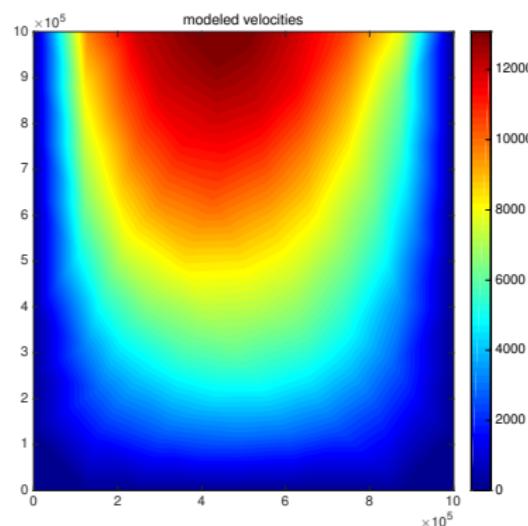
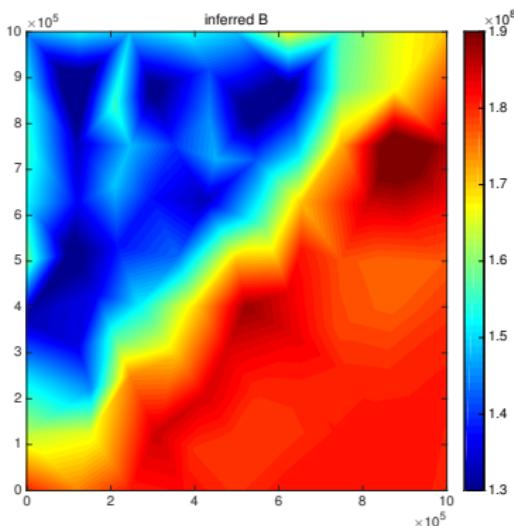


Inverse method

```
32 %invert for ice rigidity
33 loadmodel('model2.mat');
34
35 %Set up inversion parameters
36 maxsteps = 20;
37 md.inversion.iscontrol = 1;
38 md.inversion.control_parameters = {'MaterialsRheologyBbar'};
39 md.inversion.maxsteps = maxsteps;
40 md.inversion.cost_functions = 101;
41 md.inversion.cost_functions_coefficients = ones(md.mesh.numberofvertices,1);
42 md.inversion.min_parameters = cuffey(273)*ones(md.mesh.numberofvertices,1);
43 md.inversion.max_parameters = cuffey(200)*ones(md.mesh.numberofvertices,1);
44
45 %Go solve!
46 md.verbose=verbose(0);
47 md=solve(md,StressbalanceSolutionEnum());
```



Inverse method



Inverse method

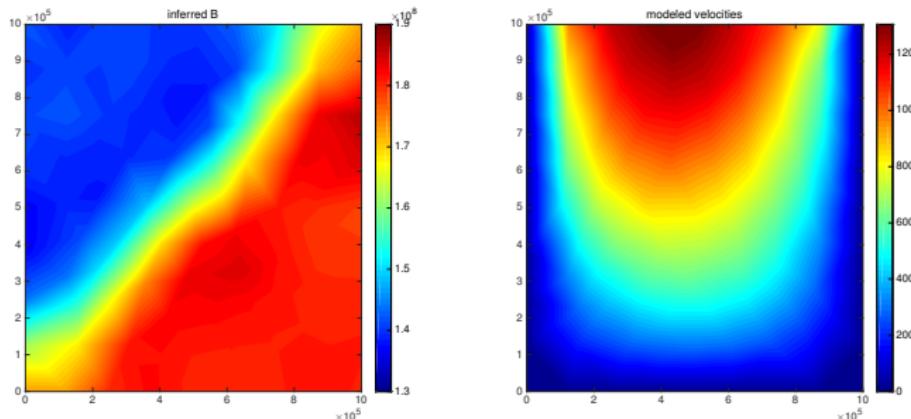
- Add Tikhonov regularization

$$\mathcal{J}(B) = \int_{\Gamma_s} \frac{1}{2} \left(v_x - v_x^{obs} \right)^2 + \frac{1}{2} \left(v_x - v_x^{obs} \right)^2 d\Gamma + \int_{\Omega} \frac{1}{2} \nabla B \cdot \nabla B d\Omega \quad (6)$$

```

60 md.inversion.cost_functions = [101 502];
61 md.inversion.cost_functions_coefficients      = ones(md.mesh.numberofvertices,1);
62 md.inversion.cost_functions_coefficients(:,2) = 10^-16*ones(md.mesh.numberofvertices,1);

```



Hands on 2 (friction)

- Same example but now for a grounded glacier:
- Changes step 1:
 - ① increase bed and surface elevation by 100 m
 - ② mask is now all grounded
 - ③ $B = 1.8 \times 10^8$ uniform
 - ④ friction coefficient: 50, and 10 for $600000 < x < 400000$
- Changes step 2:
 - ① friction coefficient uniform (50)
- Changes step 3:
 - ① We now invert for 'FrictionCoefficient'
 - ② Do we keep the same cost function (probably not...)?
 - ③ we want the parameter to be between 1 and 100

A wide-angle photograph of a desolate, icy terrain. In the foreground, a flat expanse of white, textured snow or ice stretches across the frame. Behind it, a range of mountains rises, their peaks heavily laden with thick, white snow. The mountains are rugged, with deep shadows in the valleys and bright highlights on the ridges, creating a sense of depth and scale. The sky above is a clear, pale blue, with no visible clouds, suggesting a cold, dry environment.

Thanks!